Approaches to Nesting for WRF-LIS Coupled System

John Michalakes

Mesoscale and Microscale Meteorology Division
NCAR Earth System Laboratory
National Center for Atmospheric Research

Report to Sponsor Under Task 3.8, LIS Development

Weather Research and Forecasting (WRF) Model Improvement and Support
Forecasting and Modeling (F&M)

NCAR Proposal 2009-043
MIPR F3HRA10998G004

May, 2010

I. Introduction

The purpose of this document is to explore technical approaches, costs, and benefits for implementing nested coupling between WRF and LIS. Following the "WRF-LIS Coupling via ESMF" plan (Henderson, 2007), two-way coupling has been implemented between a single domain of WRF and LIS using ESMF. This implementation supports off-line WRF-LIS nesting whereby each coupled domain in the nested scenario is run separately. However, the sponsor has migrated to on-line one-way WRF nesting for operations. The aim is to extend WRF-LIS coupling to on-line nesting in WRF.

There are a number of issues to consider. What does it mean to couple multiple domains at varying resolutions with LIS, which does not itself have nesting? Do the WRF nests each interact with a separate instance of LIS running at the corresponding WRF domain's resolution? Or do the multiple WRF nests each interact with a single instance of LIS? If so, what grid is the LIS component using and, for each WRF nest, how is interpolation handled and at what coupling interval? How is this engineered, taking into account available options for going forward as well as the effort expended so far with the current approach? Interpolating LIS data between nested WRF domains might be performed by coupling code added to the system, or it might be handled by the nesting mechanisms already in the WRF software infrastructure. And at the infrastructure level, which of the coupling libraries that have been demonstrated with WRF provides ease of implementation and maintenance of the system, ease of use for the sponsor, and reuse of work that has already been invested in this project? Each of these options will be discussed in greater detail in the sections that follow.

This report summarizes the options for implementing WRF-LIS nesting with coupling. The final section discusses lessons learned from the WRF-LIS coupling effort.

II. Nesting in WRF

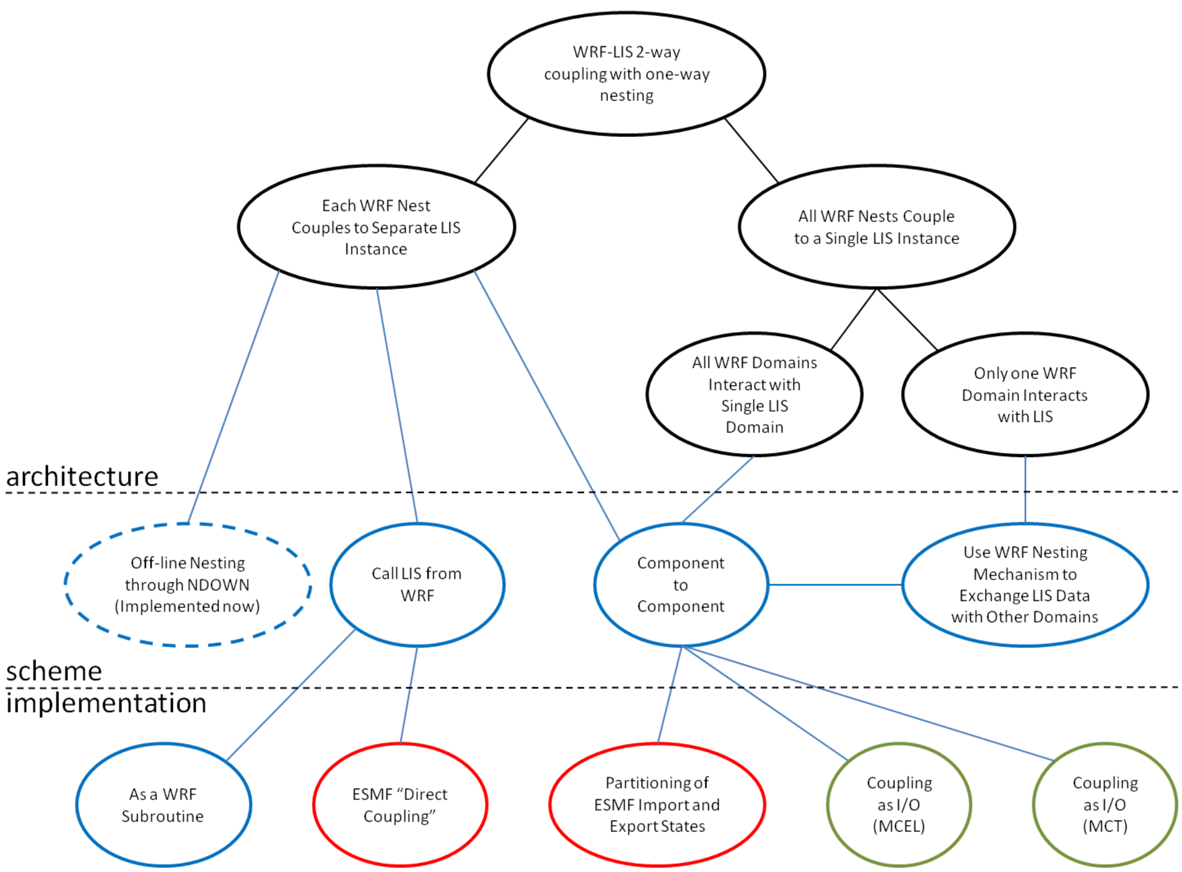
The principal difficulty this report addresses is not coupling between WRF and LIS – that has been implemented already. Rather, the issue is coupling WRF and LIS *with nesting.* Thus, it is appropriate to first describe WRF nesting and how it works, along with how coupling with nesting is more complicated than coupling a single WRF domain to another model.

A nested run of WRF comprises a set of domains running over all or part of the same geographic domain at different resolutions and time steps. For both one-way and two-way nesting, a coarser domain (parent) provides interpolated forcing data for the lateral boundaries of the higher-resolution domain (nest). The parent may provide interpolated initial conditions for the nest, the nest may be initialized from another source (input file), or from some combination of these sources. In two-way nesting (not in use at AFWA) the nest feeds back data that is then smoothed onto the parent domain. One-way nesting can be performed either off-line or on-line. Off-line, each domain is a separate run of the model. Starting with the coarse domain, output from each run is interpolated by a separate program (NDOWN) and then input to the next higher resolution run. The resolutions and time-steps between respective domains can differ by a large amount and the interval between successive parent-to-nest forcing steps can be large numbers of time steps. For on-line nesting, whether one-way or two-way, the parent-nest ratio of resolutions and time-steps must be a small integer (typically 3) and the parent and nest interact to exchange data on each parent domain time step. All domains execute as part of a single run. WRF has built-in interpolation, feedback, and time advance logic to handle the sequence of data exchanges and time steps on the parent, its nests, the nests of nests, and so on. The WRF model is very general with respect to numbers, orientations, spatial, and temporal refinement of nests.

The LIS model does not do nesting in the same sense as WRF but it does support multiple instances of LIS running in the same executable. For LIS to interact with multiple WRF domains in nested simulations there are a number of issues that must be considered. These are addressed in the next section.

III. WRF-LIS Coupling for Nesting

To design and implement a nested coupling scheme for WRF and LIS it is first necessary to consider in an implementation-neutral way what such coupling should look like, then proceed to designing coupling schemes and their implementation. One view of the problem is the decision graph in the figure below.



At the coupling *architecture* level, the choice is between multi-WRF/multi-LIS coupling, where each WRF domain interacts with its own instance of LIS, or multi-WRF/single-LIS, where each WRF domain interacts with a single instance of LIS. In the latter case, a second decision is needed: should each WRF domain interact with LIS or should they interact through a single spokesdomain; say, the main (coarsest) domain. Then, from each of these coupling architectures, coupling *schemes* and *implementations* follow. The coupling scheme options[1] are either to call LIS from WRF (probably from the surface-layer physics driver) or to couple WRF and LIS as separate components, as in the current single-domain WRF-LIS coupled system. The component-to-component option may take several forms, depending on which architecture is chosen and whether all WRF domains or only a spokesdomain interact with LIS. If coupling is between only a single instance of LIS and a single spokesdomain of WRF, the nest-forcing-and-feedback mechanisms already implemented in WRF can be used to transfer LIS data to and from the other WRF nested domains.

At the bottom level of the diagram are options for implementing the coupling. Calling LIS from WRF can be implemented either as a WRF Model Layer subroutine or using the ESMF "Direct Coupling" approach. Schemes for component-to-component coupling can be *Coupling as I/O* implemented in WRF by the Model Coupling Environment Library (MCEL) (Bettencourt 2002) and the Model Coupling Toolkit (MCT) (Jacob and Larson 2005) or by explicitly partitioning the ESMF import and export states to hold data from multiple WRF nests and LIS domains.

The potential advantages and disadvantages of these different approaches is discussed in the sections that follow.

A. Coupling Architecture

The architectural choices are couple each WRF domain to a separate instance of LIS or couple all WRF domains to a single instance of LIS. These are scheme- and implementation-neutral.

1. Each WRF Domain Couples to Separate LIS

Coupling each WRF domain to a separate instance of LIS allows each WRF-LIS coupled pair to be run on the same grid at the same resolution, eliminating the need for interpolation and loss of high resolution information. Each WRF-LIS coupled pair is free to interact on whatever interval is appropriate to the nest resolution and scales. There is additional setup overhead for running multiple LIS domains at various resolutions. There is also additional computational cost for running one LIS domain per WRF domain but this is not a major concern since LIS is lightweight compared to WRF computationally. Other difficulties of this approach are schematic- and implementation-specific rather than architectural and will be discussed shortly.

2. Each WRF Domain Couples to a Single LIS

Coupling multiple WRF domains to a single instance of LIS means that, as above, each WRF domain runs on its own grid at its own resolution and time step. However, there is only a single instance of LIS running on some grid and at some resolution. Coupling architecture issues include:

[1] Offline nesting is already implemented and shown in the figure only for completeness.

- Which grid does LIS use? The coarsest WRF domain? The finest? Its own domain entirely?
- Which WRF domain does LIS interact with? The coarsest? Finest? Each of them?
- What coupling interval between WRF and LIS?
- Interpolation is necessary between LIS and most of the WRF domains.

An advantage of coupling to a single instance of LIS is simplicity, especially from an operational point of view, since only a single LIS instance must be started and run.

B. Coupling Scheme

Coupling schemes address how the models interact and what paths data take between them. The choice is calling LIS from WRF or having the two models couple as distinct components.

1. Call LIS from WRF

A simple approach is to couple LIS to WRF in the same way other physical process models are included in WRF, as subroutines. This approach was considered relatively early in the WRF-LIS coupling project but put aside because LIS is much more a program in its own right than a simple physics subroutine. LIS performs I/O and interprocessor communication on its own, which is not allowed for WRF model layer subroutines because of the potential for breaking parallelism and thread-safety in WRF. However, the approach can be revisited.

LIS does I/O only during its initialization and spin-up phases – it does not read files when in the free-forecast part of a WRF-LIS coupled run. LIS is also more robust and better engineered than the typical physics package contributed to WRF from the user community. Thus, an exception could be made for including calls to LIS at the point in WRF solver where it normally calls other surface layer physics packages.

While not strictly a “calling LIS from WRF” scheme, having LIS and WRF interact at the surface-physics call site in WRF is also something that ESMF supports in the form of "Direct Coupling" (Thurich, 2008), though further investigation is needed determine whether this approach provides advantages over simple subroutine-coupling.

2. Component-to-Component Coupling

Here component-to-component means coupling WRF and LIS as complete components without invading the call tree of either code, as opposed to the approaches in the section above. The component-to-component scheme supports any of the coupling architectures described above, but needs additional structure in the form of developer- and library-supplied infrastructure to mediate the coupling. Regardless, and even in the single-domain case, the component-to-component coupling scheme requires the models to present pre-existing or specially engineered ports through which data may be transferred and to provide hooks allowing control of the startup and sequencing within the coupled system. The ports into and out WRF need must be either replicated or enlarged to account for the multiplicity of WRF nests exchanging data with the single or multiple instances of LIS. Routing all coupling data through a single enlarged port requires keeping track of which data coming through the port belongs to which domains. Having a port for each WRF nest to communicate to LIS makes the traffic easy to keep separate.

The component-to-component scheme has the advantage of having been already implemented for the single-domain WRF-LIS system.

3. Other Schematic Issues

Aside from the foregoing issues for the coupling scheme, there are two others: sequential versus concurrent scheduling, and single versus multiple executables.

Sequential scheduling means that the components take turns running. When WRF is running LIS is waiting and vice versa. WRF and LIS are sequentially scheduled in the current single-domain implementation of the coupled system. This is the natural schedule when one application calls another as a subroutine and can also be implemented for component-to-component coupling. Sequential scheduling is especially useful when two-way coupled models cannot tolerate being slightly out of phase in simulation time with each other. The main disadvantage of sequential scheduling is that each model runs on the full set of processors so the ability to run on large numbers of tasks is limited by the least scalable component.

Concurrent scheduling means that the models run at the same time on possibly different sets of tasks. Coupled models with different computational workloads or scaling behavior benefit from concurrent scheduling, though care must be taken to balance the number of tasks assigned to each to avoid load imbalance. If the coupling is two-way, then one model must be lagged one coupling interval behind the other to avoid deadlock or serialization. A small phase lag is usually not an issue as long as the coupling interval is short enough, nor is it likely to pose an issue for WRF-LIS.

Either sequential or concurrent scheduling is appropriate for WRF-LIS, though one may be more workable than the other depending on the particular architecture, scheme, and implementation. Likewise, whether WRF and LIS execute as a single or multiple executables is significant as an implementation issue, particularly for integration, maintenance, distribution, and support of the codes and their build mechanisms.

C. Implementation

This section describes the implementation schemes diagrammed across the bottom row of the figure. From the left, the first option is coupling LIS as a WRF Model Layer subroutine. Next is an option for ESMF coupling that can be used to call LIS as a subroutine. The last three options are ways to implement component-to-component coupling. One partitions the ESMF import and export states being passed into WRF to allow them to contain data for multiple nests – this is analogous to making the application's coupling port bigger. The other two use the coupling-as-I/O approach, analogous to replicating the coupling ports. These are implemented using infrastructure already included in WRF, one using the Model Coupling Environment Library (MCEL) and the other the Model Coupling Toolkit (MCT).

1. Overview of Coupling Packages

ESMF is a library that includes model coupling functionality, as well as a framework for constructing new model components or systems of components. It has been used extensively in the NASA GEOS 5 model and forms the basis for the NCEP Environmental Modeling System (NEMS) being developed and the Common Model Infrastructure recommendations of the National Unified Operational Prediction Capability (NUOPC) initiative (www.weather.gov/nuopc).

MCT is a library that provides parallel communication and interpolation of data between component applications running on the same or different sets of processors. It supports both single- and multi-executable coupled systems and concurrent and sequential (as well as hybrid) scheduling of the components. MCT is the basis for the coupler in the Community Climate System Model (CCSM) and has been used to couple WRF to the ROMS ocean model.

MCEL was developed for use at Naval Research Laboratory under DoD HPCMO funding, and has been used to couple WRF to the HyCOM and ADCIRC ocean models and other models. MCEL is supported by a single person, the original developer. It is based on the CORBA client-server process model. The MCEL server is a single process, which may be multi-threaded up to the number of processors on a node, but that is the limit of its scalability. Similarly, the communication channels between the server and the client model components are implemented as a single pair of TCP/IP sockets per pipe, so the communication pathways are also not scalable. However, with 2D data going between components running on modest (hundred or fewer) numbers of MPI tasks, the coupling overhead is competitive with ESMF or MCT, which do support scalable parallel interpolation and data movement.

2. Calling LIS as a WRF Subroutine

WRF incorporates many instances of physics for convection, microphysics, radiation, boundary layer, surface processes, and others. For reasons discussed above, coupling LIS as a subroutine was considered but put aside in the face of other potential complications for WRF parallelism and software architecture. An advantage for revisiting this approach is that it supports nested coupling trivially, at least as far as the WRF computation is concerned. Land surface processes are called from within the solve routine which computes one time step on one domain. Each WRF domain maintains its own set of state data that would be transferred to LIS through the argument list of the subroutine call in the invocation of the solver for that domain. With the multiple-WRF/multiple-LIS coupling architecture, there would need to be separate instances of LIS, each initialized and running with its own set of state data. LIS supports multiple instances of itself in the same executable. With the multiple-WRF/single-LIS architecture, each invocation of LIS from a different WRF nest would need to include mechanisms for interpolating between the LIS grid and the grid of the WRF domain calling it. This would increase the engineering effort for implementing nesting, but could make use of some of the nest forcing and feedback infrastructure already in WRF. It would also be relatively straightforward to have this part of the code generated by the Registry program (Michalakes et al. 2005).

3. Direct Coupling through ESMF

This approach has many of the advantages and disadvantages of the "LIS as WRF subroutine" approach. From the WRF point of view, nesting is still trivial since the call site to the LIS ESMF coupling is within the solver that is called on each time step for each domain. For the multiple-WRF/multiple-LIS architecture, Direct Coupling through ESMF is overkill, since the WRF-LIS pairs can run on the same grid. Direct Coupling duplicates what's already available from the WRF Model Layer Interface and a Fortran subroutine call. The value of the Direct Coupling approach would be for the multiple-WRF/single-LIS architecture, since ESMF could handle interpolation between WRF and LIS at each invocation. And it is reasonable to expect parts of this code can be generated by the Registry too.

4. Partitioning of ESMF Import and Export States

This implementation extends the ESMF component-to-component coupling that has been implemented for the single-domain WRF-LIS. In the single-domain coupled WRF-LIS system, each component plus a coupler component are called within a top-level loop, executing one time period on each application with exchange of data through the coupler each time through. When WRF or LIS are invoked, they are passed two data structures called ESMF States, one for import (data coming into the application) and one for export (data coming back out of the application). The applications themselves handle unpacking and packing the data from the ESMF States into and out of their own data structures. The situation becomes more complex with nesting, since there must be a set of import and export state data for each WRF nest, but there is still only one state in and one state out of the application at each coupling step. Further complicating the situation is the fact that the data comes in from the top-level of the code. It is up to the application itself to work out the partitioning of the import and export states and then routing those to the appropriate nest at the appropriate time in the calculation. It is possible to do this in WRF (anything is possible with enough effort and patience) but it is the most effort intensive option of the group, especially considering that it must be done in a way that is compatible with the existing WRF nesting scheme and software architecture.

WRF nests may telescope to arbitrary levels and they may be created, destroyed, and moved at run-time according to specifications in the WRF namelist. Any partitioning of the ESMF import and export states to accommodate this, as well as the code in WRF to unpack, pack, and manage the transfer of the data to the correct domain would also have to be this general and flexible. Although some of the groundwork is already available in the single-domain WRF-LIS system, designing, developing, and testing such an implementation for nesting would require considerable effort by a skilled software engineer intimately familiar with the WRF nesting and other aspects of the infrastructure. A quick-and-simple hard-coded partitioning for a particular set of nested domains will not meet the needs of the sponsor or the WRF community at large.

5. Use of ESMF Exchange Grid

The complexity of partitioning ESMF states, above, may be avoided by using a an ESMF Exchange Grid, which runs LIS at the finest WRF resolution but over the entire outermost WRF domain. The import and export states to and from WRF contain this single comprehensive fine-scale grid (the exchange grid), avoiding the need for partitioning the states. This approach is being explored for moving-nest coupling of COAMPS and WRF to HYCOM and the Princeton Ocean Model for hurricane prediction by Tim Campbell at Naval Research Laboratory (Campbell, 2009) under NOPP funding (Chen and Ginis, 2009). It is explained in more detail in the Tradeoffs section, below.

6. *Coupling as I/O*

Coupling as I/O is a natural approach to managing the transfer of data between multiple domains in the WRF model and multiple instances of LIS is through the WRF I/O and Coupling interface in the WRF infrastructure – essentially handling coupling from multiple domains the same way WRF handles I/O to files from multiple domains. This widely used paradigm forms the basis of the PRISM (Guilyardi, 2002) and the Models-3 (Coats et al. 1999) coupling modeling systems. Coupling as I/O is built into WRF and has been demonstrated coupling WRF to the HyCOM, ROMS, and ADCIRC ocean models as well as a CFD-based urban air flow model. Each domain in WRF has the ability to open I/O streams using various formats such as NetCDF, Grib, HDF, and unformatted binary. Each format mechanism is an implementation of the WRF I/O and Coupling API. Similarly, WRF can read and write formats that implement coupling to another application instead to a file. The model sees nothing different. The Coupling as I/O mechanisms implemented in WRF are MCT, MCEL, and ESMF.

For the MCT and MCEL implementations, coupling-as-I/O solves the multiplex problem into and out of WRF without additional engineering to partition or route the data. In the case of ESMF, coupling works for a single domain, not multiple nests, because although the WRF infrastructure makes it appear to the model as coupling-as-I/O, coupling data is actually passed as import and export states through single port at the top of the model code, the ESMF component interface. WRF infrastructure and API mechanisms would need to be reengineered to support nested coupling using ESMF.

D. *Tradeoffs and Suggested Next Steps*

Each of the options for nesting in the WRF-LIS coupled system presents advantages and disadvantages which must be weighed in the context of the system goals and requirements:

- The system is intended for operational use at AFWA. It must fit into the current operational environment, meet real-time performance requirements, and be usable and manageable by center personnel.
- The system must provide needed functionality and added value by improving forecasts both by itself and as part of a larger ensemble forecast system.
- The sponsor also intends that the WRF-LIS coupled system can become part of the officially supported WRF code in some form, and made available to the WRF user community.
- The implementation and of the system must be affordable.
- The system must be maintainable by the WRF and LIS developers on an ongoing basis.

Suitability to requirements and tradeoffs for going forward are presented as follows:

1. *Couple LIS to WRF as a Model Layer Subroutine*

This was one of the original approaches considered for the single domain system. LIS would be added as a Model Layer subroutine to the WRF code as another option for surface layer physics. LIS could be added to the WRF build or maintained as a separate code that appears as a library. This approach discards the last two years work on the single-domain WRF LIS system. It does not require or use ESMF. Attention would be needed to ensure LIS parallelism and I/O works correctly without interfering with WRF parallelism or thread-safety. Since nesting is already implemented for packages included in WRF as subroutines, coupling multiple WRF nests to LIS in this way is straightforward on the LIS side. Estimated engineering effort: 0.5 to 0.75 FTE.

2. *Couple LIS to WRF using ESMF Direct Coupling*

This is similar to the previous approach. Coupling to LIS is invoked for each domain-step from within WRF at the surface driver interface in the solver; thus nesting is already provided for on the WRF side. This approach also discards the previous work on the coupled WRF-LIS system, but preserves a role for ESMF, interpolating when multiple WRF nests couple to a single LIS domain. If each WRF domain is coupled to a separate instance of LIS, interpolation is not required and the need for ESMF Direct Coupling is obviated. Further investigation and help from the ESMF core team are needed for the Direct Coupling ESMF approach. Estimated engineering effort for Direct Coupling multiple WRF domains with a single LIS instance: 0.75 to 1 FTE.

3. *Add multi-domain coupling to Current System*

This option extends the single-domain ESMF-coupled component-to-component system developed over the past two years to allow coupling of multiple WRF nests to LIS (either single or multiple LIS instances) through the import and export state arguments of the WRF component interface to ESMF. This approach leverages and builds upon the existing WRF-LIS system and is ESMF "compliant."

One approach involves partitioning the ESMF import and export states. It would be the most difficult and expensive to implement in a way that is general and conformable to the WRF nesting and I/O infrastructure. Regarding maintenance, although we expected to see advantages of modularity and interoperability from coupling WRF and LIS as separate components, these have not been realized in the single-domain WRF-LIS system (see Lessons Learned, below). This will be worse when the complexity of nesting is added. Estimated engineering effort: 1.0 to 1.5 FTE.

4. *Add Exchange Grid to Current System*

Implementing an exchange grid under the Component-to-Component scheme, the ESMF implementation of the WRF I/O and Coupling API would need to be reengineered to implement operations such as per-nest interpolation and feedback on the exchange grid. Care must be taken to avoid introducing WRF-LIS specific data conversion algorithms into the WRF I/O and Coupling infrastructure, which would violate separation of concern and modularity in the WRF design. These would still need to be handled by a coupler. The LIS model would require no additional engineering for nesting, but would need to be run at the most costly resolution over the largest domain. Estimated engineering effort: 0.75 to 1.0 FTE.

Implementing an exchange grid combined with the ESMF Direct Coupling approach would be simpler and would avoid the need to modify the WRF I/O and Coupling Infrastructure. This avoids reengineering on the LIS side to handle interactions from multiple WRF domains, which would be managed in the Direct Coupling interface. Estimated engineering effort: 0.5 to 0.75 FTE.

5. Add multi-domain coupling using Coupling as I/O

The MCEL or MCT implementations in WRF support true coupling-as-I/O on a domain-by-domain basis so coupling in this way for nesting is already implemented on the WRF side and easy to implement on the LIS side. Considered at the outset of the project, these would have been straightforward implementations of WRF coupling when it came time to consider on-line nesting.

MCT is robust and well supported. It is also scalable to large numbers of processors. Using MCT, additional coupler logic would be needed to handle the multi-WRF/single-LIS architecture. Estimated engineering effort is 0.5-0.75 FTE.

The MCEL server can interact with arbitrary numbers of coupled components and will work for nesting out-of-the box. MCEL is not scalable but provides good performance for components running on up to a hundred or so MPI tasks. MCEL is supported by a single person, the original developer. Estimated engineering effort is 0.25-0.5 FTE.

6. Add Single-Spokesdomain Coupling to Current System

This approach couples LIS to only one WRF domain, probably the coarsest main domain; the coupling data is interpolated to and fed back from the other nested domains internal to the application using the existing WRF nesting mechanisms. The coupling is ESMF component-to-component, as with the current system. Investment in the current system is preserved, since it already supports coupling between one domain of LIS and one domain of WRF. Most of the system for nesting is already built. The main challenges of this approach are more scientific than engineering: can the system still add forecast value when fine-scale coupling information from the WRF nests is lost in the exchange with LIS? If such loss is acceptable, the engineering effort required is less than 1 month FTE.

Recommendations

The following are the author's own at time of writing. The above options should be considered and decided based on cost of implementation, ease of maintenance of the system, suitability for operational use by the sponsor and adoption by the WRF community at large, reuse of work already invested in this project, and any other needs or requirements known now or that emerge. By far the most straightforward approach is to use the off-line nesting capability that is already implemented for the WRF-LIS system. The next easiest approach is couple LIS to only the top level WRF domain and use the WRF nesting mechanism to update the nests with LIS-modified coarse domain fields (option 6). After these, other favored approaches are to utilize either the coupling-as-I/O capability already in WRF (option 5) to couple each WRF nest to its own instance of LIS through MCT or the ESMF Exchange Grid approach (option 4), which has the advantage of leveraging promising development underway at NRL/Stennis for the NOPP Hurricane Forecast Improvement project. The recommendation is to use option 6 on an interim basis while either options 4 or 5 are developed.

IV. Lessons Learned

The component-to-component coupled scheme used in the single-domain WRF-LIS system would seem to provide better abstraction and encapsulation between the two separate development efforts. The resulting system should have been easier to develop, work with, and maintain as two separate programs. This was not the case in practice. First, the tinker-toy notion of coupling two complex applications is simplistic. There isn't really a single well-defined rod between two wooden spools. Considerable effort was spent on reconciling WRF and LIS notions of land mask and sea-ice coverage, for which the coupling infrastructure was of no use. Second, in spite of the apparent top level simplicity of the coupling interface between LIS and WRF, incorporating the externally generated surface fields from LIS into the WRF was difficult because both WRF and LIS support multiple options for surface schemes and the sets of fields exchanged differ from pairing to pairing. Third, threading new fields from LIS through the WRF call tree was complex. All of these were sources for error, confusion, and costly delay. In retrospect, either the a "Call LIS from WRF" scheme or a true "Coupling as I/O" scheme would have made the single-domain WRF-LIS system easier to develop, test, and maintain; nor would adding nesting now be so difficult.

More generally, from a project management point of view, the a priori mandate that ESMF be used to couple WRF and LIS short-circuited the software design process by placing implementation before requirements gathering. Benefit of hindsight notwithstanding, it is obvious now that a great deal of time and effort was spent trying to make a predetermined solution fit the problem. It is unlikely the project would now be at this crossroad regarding nesting if the foregoing discussion in this report had been conducted at the beginning of the project.

Lastly, an important lesson learned is a fuller appreciation of the inherent difficultly of two separate groups, with only partially overlapping objectives, collaborating at distance to design and develop a coupled system from two separately developed applications. Regular and constructive communication is essential. To the extent that software technology is the issue, the early establishment of a jointly accessible development repository along with mutually agreed policies and procedures for administering the code base saves enormous amounts of thrashing over version issues.

V. References

1. Henderson, T., J. Michalakes, F. Chen, J. Dudhia, S. Kumar, and C. Peters-Lidard. WRF-LIS Coupling via ESMF. Report to sponsor. September 4, 2007.
2. Thurich, Gerhard. ESMF Presentation: www.earthsystemmodeling.org/presentations/pres_0804_directcoupling.pdf

3. Coats, C. J., A. Trayanov, J. N. McHenry, A. Xiu, A. Gibbs-Lario, and C. D. Peters-Lidard, 1999: An Extension of The EDSS/Models-3 I/O API for Coupling Concurrent Environmental Models, with Applications to Air Quality and Hydrology, Preprints, 15th lIPS Conference, Amer. Meteor. Soc. Dallas, Tx., January 10-15, 1999
4. Guilyardi, E., R. Budich, G. Brasseur, and G. Komen, 2002: PRISM System Specification Handbook V.1. PRISM Report Series No 1 239 pp.
5. Larson, J., R. Jacob, and E. Ong. The Model Coupling Toolkit: A New Fortran90 Toolkit for Building Multiphysics Parallel Coupled Models. International Journal of High Performance Computing Applications, 19(3), 2005, pp. 277-292.
6. Bettencourt, M. Distributed Model Coupling Framework. In Proceedings of the 11th IEEE Symposium on High Performance Computing. 2002. pp. 284-290.
7. Michalakes, J., J., J. Dudhia, D. Gill, T. Henderson, J. Klemp, W. Skamarock, and W. Wang: The Weather Research and Forecast Model: Software Architecture and Performance. Proceedings of the Eleventh ECMWF Workshop on the Use of High Performance Computing in Meteorology. World Scientific, 2005, pp 156-168
8. Campbell, Tim. Naval Research Laboratory, Stennis Space Center. Personal communication.
9. Chen, S. and I. Ginis. A Unified Air-Sea Interface for Fully Coupled Atmosphere-Wave-Ocean Models for Improving Intensity Prediction of Tropical Cyclones. Joint proposal to ONR in response to BAA 09-012 (funded).